Network Security Challenges and Countermeasures in SDN Environments.

Abdelrahman Hegazy* and Minar El-Aasser†
Information and Engineering Technology, German University in Cairo
Cairo, Egypt
Email: *abdel-rahman.hegazy@student.guc.edu.eg, †minar.elaasser@guc.edu.eg

2021 Eighth International Conference on Software Defined Systems (SDS) | 978-1-6654-5820-7/21/$31.00 ©2021 IEEE | DOI: 10.1109/SDS54264.2021.9732104

***Abstract*—The internet has made everything connected and accessible from anywhere. However, conventional TCP/IP networks are complex and very hard to manage. Software-Defined Networking (SDN) is one of the most promising networking paradigms in current and next-generation networks. It promises to change this situation by breaking vertical integration and introduce network programmability. SDN separates control from the network and data plane. The intelligence and brain of the network are logically centralized, and the underlying network infrastructure is abstracted from the application. However, the Control Plane and Data Plane separation opens the door for security challenges and threats. In this paper, we aim to collect, analyze and classify all major security threats and their possible solutions. The security platforms that are used as countermeasures for each attack are described, followed by various security approaches for network-wide security in SDN. As well as classifying security challenges and threats according to different fields, an SDN simulation platform to study and test network performance and attacks countermeasures is also introduced. In short, this paper gathers all the present major SDN security challenges and possible solutions. Furthermore, it studies, classifies and highlights future directions for secure SDN.**

KeyWords: Software-Defined Networking, Network Security, SDN Security Threats, Security Countermeasures, OpenFlow.

I. INTRODUCTION

Recently, Software-Defined Networking (SDN) has been regarded as one of the most popular network paradigms for next generation networks. It has become one of the most important network architectures to simplify network management and realize communication network innovation. In traditional network architecture, most network functions are implemented on dedicated equipment and hardware i.e. the switches, routers, etc. [10] and operating and maintaining current networks is a difficult task. Implementing a new global strategy or changing a small group of devices requires individual configuration of each device, which makes the task complex and time-consuming.

SDN has the potential to reduce many of these traditional network problems because it supports the dynamic nature of the network, the centralized control plane, and the ability to direct programmability. SDN relies on the separation of the network control plane from the data forwarding plane. Due to the distinctive feature of data plane and control plane separation, SDN provides an efficient solution to support diversified network functions, dynamic network management, and network security as well as simplifying policy enforcement and network (re)configuration and evolution [6].

The separation of the control plane and the data plane can be accomplished through a well-defined programming interface between the switch and the SDN controller. Currently, OpenFlow is the most widely accepted and used SDN implementation architecture. In OpenFlow, network policies and services are implemented as OpenFlow applications, which interact with the control plane through APIs (Application Programming Interfaces) [13].

SDN has its own challenges and limitations in terms of security, scalability, and compatibility. Security has always been at the front of these challenges. Security vulnerabilities in the communication of the controller data path can lead to illegal access and use of network resources. If the security of SDN cannot be guaranteed in the process of replacement of the traditional network architecture, its development will meet great resistance and even become completely irrelevant. Although research and implementation of SDN have made great progress, research on SDN security is still in its infancy. Researchers tried to test and analyze the performance of SDN networks and controllers, as well as trying to identify possible threats to and attacks on SDN networks.

The main aim of this paper is to gather all the work of researchers and scientists in the field of SDN network security. Our goal is to provide a comprehensive and up-to date SDN security overview by presenting security challenges and possible solutions. We aim to gather and classify all the attacks discussed with all the possible solutions. Additionally, we aim to facilitate discussions on SDN security, and gather all possible data in the scope of that topic in one place, so that research scientists can have both a starting point and a reference of previous work in this field. We also aim to introduce a simulation of the developed SDN platform, which can be used to test multiple topologies of SDN architecture.

This paper is organized as follows: Section II introduces a background about SDN and traditional networks followed by Security challenges with their possible solutions of SDN in Section III. A proposed classification for threats and possible solutions is presented in Section IV. The developed OMNeT++ SDN platform is presented in Section V. Future research directions for security in SDN are outlined in Section VI and the paper is concluded in Section VII.

II. BACKGROUND

In this section, we review the background of SDN and the major differences between SDN and conventional networking.

A. *Conventional Network Architecture.*

Traditional IP networks are very complex and hard to manage. Configuring the network according to predefined policies and re-configuring it to adapt to faults, load, and adjustments are all challenging tasks.

What makes the traditional IP networks that difficult to handle is that the network is vertically integrated with the management plane, control plane, and data plane which are packed together, as shown in figure 1. The management plane is responsible for the way to configure a network device, whether by physical cable through the device itself or using one of the management protocols such as telnet, SSH, or SNMP. The control plane is the brain of the network device and the configuration that's written in the terminal to run the network such as routing protocols and addressing tables. And finally, the data plane which is the hardware part of the network, that applies the configurations written in the control plane. This plane is responsible for forwarding the packets following the forwarding tables and routing protocols from the control plane.

Among the ordinary network hierarchy, it uses integrated hardware and software to direct traffic across a series of routers and switches. This process takes a lot of time and effort, as it configures one or multiple end devices in your network, or adds new devices to your current network criteria. This shows both complexity and network burden that open the way for human errors and limitations of innovation and automation, this is because each hardware device for a specific vendor is committed to a specific software and method of configuration. Also, it suffers from a large conversion time when reporting a node failure or a problem in one of the addressing tables.

B. *Software-Defined Networks (SDN).*

SDN has been recently identified as a new communication paradigm in computer networks; it serves as a different perspective when looking at a network. People around the world nowadays call SDN "a network in your computer". SDN changes what happens on the control plane. By using SDN, there will be flexibility and agility using your network as well as less complexity than traditional networks [2].

SDN separates the service of network devices into a control plane and a data plane. The control plane serves as a central unit controlling the network. While the data plane serves as a plane to route data by flow entries in the switch flow table to allow for the successful delivery of the packet. SDN focuses on control plane programmability and the target is to implement a unified centralized network architecture. SDN's goal is to enable network engineers and administrators to respond quickly to changing business needs. Since there is a software layer that controls the hardware underneath, applications can be built on the software layer to access and control external network resources [2][1].

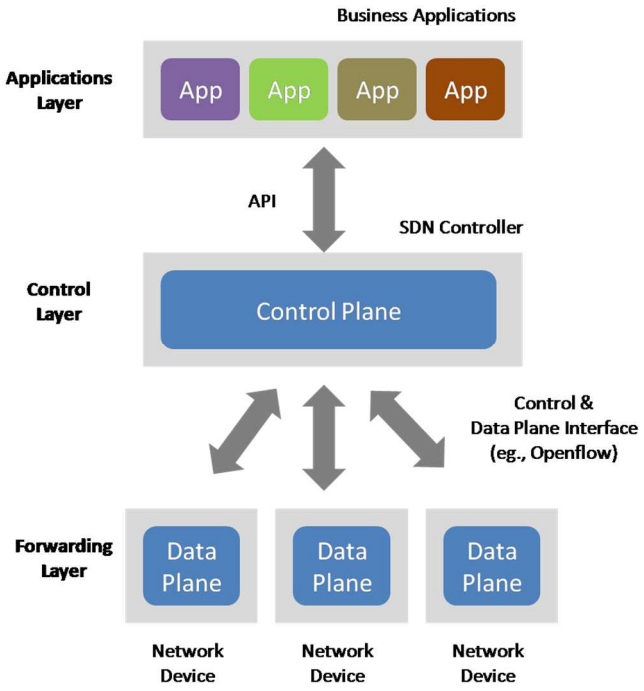

Fig. 1. SDN network architecture [27]

Control Plane. In SDN, the control plane is taken from individual network nodes and implemented in another logically centralized plane. The control plane is the plane responsible for managing and controlling the network. This happens through adding a new device to the network. The controller is a centralized device to think and serve as the brain for all the network devices. This intelligent controller runs a specialized software that manages all the network traffic in the data center and a series of routers and switches that forward packets of traffic. The controller also extracts information about the network from the hardware devices and communicates back to the SDN applications with an abstract view of the network, including statistics and events about what is happening [2][5].

The control plane communicates with two different planes in the network. The first plane is the Application Plane above the controller in the hierarchy. That plane contains Application Program Interfaces (APIs) which are responsible for the open-source applications used to program the controller. Then, these APIs communicate with the controller using OpenFlow protocol. The second plane the control plane communicates with is the Data plane. The controller puts the logic that controls forwarding behaviors, this Includes routing protocols and configuration for network devices. The controller puts the data configures in a table called the Flow Table, then passes the table to all network devices using OpenFlow protocol [1].

Data Plane. SDN's separation of control and data planes means simplifying transmission devices and allowing them to be controlled remotely via an open interface. The data plane is where traffic flows and entrusts packets forward to straightforward switches. The SDN networking devices

control the forwarding and data processing capabilities of the network, this includes the forwarding and processing of the data path according to control plane logic. Data plane functions include forwarding packets at the IP layer, as well as switching at layer two. So, the routing protocol function that computes the paths is the control plane function. While the act of taking a packet on an input port and forwarding it to an output port is a data plane function. The most appropriate example of an SDN transport device is an OpenFlow switch [1]. OpenFlow switches have a future because the simple transport policy is enforced by the controller software rather than the switch's hardware and firmware. By communicating with the control plane using OpenFlow protocol, the control plane passes the flow table after finishing its routing protocols. Then the data plane makes lookups in the flow table to determine how to route the packet. By checking the flow table entries, such as source address, a destination address, or destination ports, the data plane decides what to do with the packet and then it routes the data [11].

C. *OpenFlow.*

The OpenFlow protocol is a standardized protocol for interacting with the forwarding behaviors of switches from multiple vendors. This provides us a way to control the behavior of switches throughout our network dynamically and programmatically. Openflow is a key protocol in SDN solutions. It is the protocol responsible for communication between the controller and the network devices. It is responsible for distributing the flow table to all network devices. Additionally, the network devices report any errors or failures to the controller using it. The OpenFlow controller provides a clear integrated view of your network; you can find network vulnerabilities and intrusions more easily and implement security policies [14].

SDN introduced the network idea of being open across Vendor interoperability and programable, the controller deals with a higher-level Application Programming Interface, these APIs allow the programmability of the controller using open-source applications such as Python and Java. OpenFlow also increases the performance of the network by providing security and easy management. The OpenFlow switch doesn't process the data as the conventional switches, it works based on the OpenFlow table involving only the data plane [5].

D. *Security in SDN.*

Virtualizing the network comes with advantages: the network can be separated up and down dynamically, they can be fine-tuned for specific application use; cases and security policies can be installed on each network. It also opens up to fast responses to all network failures which results in less conversion time. It is also better at avoiding loops, having simpler management, and supporting automation and innovation across vendors. This happens because the controller software is open, written by open source applications as well as opening the way for a flexible network to control. Despite coming with many advantages, virtualizing the network also comes with challenges, which hinder the implementation of SDN. The main challenges are performance, scalability, security, and interoperability. In this paper we will focus on the security challenges in implementing SDN and how can we protect SDN from malicious attacks [26].

SDN Security Challenges are divided into two scopes: Challenges on the outer level of the network and others on the inner level of the network. The outer level challenges target the data plane of the architecture, which is the Services and Switches of the network, this happens by TCP-Level attacks, flooding attacks (DoS attacks), and the Flow Tables memory limitation. The inner level challenges target the Application Plane and Control Plane, which are the controller and SDN Interfaces. This could happen through network manipulation and hijacks, controller compromises, or lack of Authorization and Authentication [22] [1].

III. SDN THREATS AND POSSIBLE SOLUTIONS.

The separation of the control plane and data plane and aggregate control plane functions in a centralized system are important to future networks. Although it has a lot of advantages to the SDN networks, it also opens new security challenges. The control plane is the most attractive part of the SDN network for security attacks, this is due to its importance to the network, as it has the controller which runs everything in the network. The SDN controller can become a single point of failure and bring the entire network down if security is compromised. This happens due to resources visibility which is important in SDN, but it should not be visible to all or unconcerned applications [1].

Over the years, the list of the security challenges and attacks were expected to grow with the gradual deployment of SDN technologies. To take full advantage of SDN, these challenges need to be emphasized so that appropriate security measures can be proactively taken. Therefore, security challenges and threats existing in SDN are discussed in detail in this section, alongside possible mitigation techniques and network solutions. And also in this section, we will specify the attack points in each attack, in order to know which part of the SDN network topology is associated with each specific attack [22].

A. Lack of Authentication and Authorization.

One of the major issues in SDN domains is applications' lack of authentication and authorization. In OpenFlow, the controller runs applications that are responsible for implementing a majority of the functionalities of the control plane. These applications are developed by different parties other than the controller vendors. These applications have inherited access rights to network resources and manipulation of network behavior, and most of them do not have adequate security mechanisms to protect network resources from malicious activity. So, the authentication of growing applications in a programmable network using a centralized control architecture is a major security challenge. This type of security threat is associated with the Application layer. There is a lack of mechanisms to establish trust between controller and management applications [28] [1].

Possible Solutions:

Limiting the number of applications used in programming the functionalities of the SDN networks through the controller.

The FRESCO scripting language enables developers to implement new security applications which can be deployed on any OpenFlow controller or switch implementation [25].

B. Lack of Access Control and Accountability.

In SDN, most network services are implemented using applications. This means that adequate access control and accountability mechanisms are required to ensure the security of the network. There are three classes of applications that can affect network security in SDN. The first one is network sensitive applications; these are applications that require certain network characteristics, such as: accessing ports VLANs, monitoring costs of traffic and flows, etc. The second class of applications is applications that provide services to the network, such as: access control or firewall, and content inspection or intrusion detection services. Thirdly, Packaged Network Services applications, this class combines the previous two classes; an application that requires instantiation of another application as a virtual element of the network, such as pushing traffic through firewall applications. So, a malicious application can pass wrong access control by using an instance of another application.

Possible Solutions:

Applications need to work within their functional boundaries and have controlled access to network resources. Most defenses for SDN related to access control are Role-based access control (RBAC) [1].

SE-floodlight implements a secure application kernel layer to mediate flow rule requests between different roles [20]. Alongside Rosemary, AEGIS, SDNShield, and SM-ONOS [17] [29] [31].

C. Malevolent Applications.

Applications deployed on top of the control plane may pose a serious security threat to the control plane. From the perspective of the controller's ability to verify the resources used by applications and authorized applications through proper isolation, auditing, and tracking, controller security is challenging. Besides that, before providing access to information and network resources, different applications need to be separated according to their security implications. Therefore, the controller's northbound API requires a custom security execution mechanism for various applications. This custom security program based on the type or category of the application has not been tested.

Possible Solutions:

SE-Floodlight [20] provide northbound programmable API to handle usage rights and act as an arbiter between the application and the controller.

FRESCO [25] is a development system build for the security of the application. This platform is also used in protecting the SDN networks from malevolent applications.

D. Rules Insertion Manipulation.

In the application layer, if an application is compromised it can generate false flow rules. These false flow rules will fill the flow table and compromise the network. If an attacker controls certain applications that are used to program the controller flow table insertion rules, the whole network can be compromised, and it is difficult to check if an application is compromised or not [1].

The data plane also must be protected from malicious applications that can install, change, or modify the flow rules on the data path. So, this type of security threat attacks the application plane alongside the data plane and links between the application plane and control plane. This type of threat also attacks the flow table, as the flow table can be a victim to false flow rules from the higher applications [21].

Possible Solutions:

Platforms that allow OpenFlow controllers to check flow rule contradictions in real-time and authorize them before the OpenFlow application changes the flow rules, such as: FortNox, FlowChecker, and VeriFlow.

E. Scalability and Availability.

Controllers cannot handle the huge number of new flows using OpenFlow. It is described as a lack of scalability, which enables targeted attacks to cause control plane saturation. So, controller scalability makes it a favorite choice for DoS and DDoS attackers [1].

The controller can be easily considered as a bottleneck as it's required to install flow rules for each new flow in the data path. Delay constraints are another challenge facing SDN controllers. This happens if the number of forwarding devices is too huge to be managed by a single controller. If the number of flows increases, the possessing time will increase, which depends on the controller's processing power [1].

Possible Solutions:

The distributed SDN control plane (DISCO) [18] [19].

Increasing the processing power of the controllers is one of the solutions to overcome the problem of scalability and availability [7].

Hybrid controller architecture with the intelligence to act in advance allows the controller to act responsively to set the path, understand the behavior of traffic in advance, and define the path.

F. Flooding Attacks.

Flood attacks allow attackers to disrupt the normal operation of the SDN network by causing them to downgrade or deny the expected level of service. When the input rate to be processed exceeds the processing capacity, the flooding will cause system failures. Flooding attacks are not that far off from the DoS and DDoS attack. In SDN, the controller calculates and inserts the flows of the network in the flow table, which can store a finite or limited number of flow rules [21]. Flooding attacks lead to denial of services (DoS). Flooding in the data plane can cause the control plane to stop working properly. The data plane is vulnerable to saturation attacks because it has limited resources to buffer unsolicited (TCP / UDP) flows.

Possible Solutions:

Security softwares and platforms aim to overcome flooding attacks, such as: SWGuard[32], LineSwitch[3], FireGuard[30], and Avant-Guard[24].

G. *DoS and DDoS Attacks.*

DoS and DDoS attacks are the most common threatening attacks in the field of networking in general, and also in SDN. DoS attack is the idea of making network resources not available to its regular users. In the paper [23], a DoS attack is demonstrated. Network inspection tools are being developed that can identify SDN networks with the help of flow response times. The data path for the new flow has a different flow response time for the new flow and the old flow to execute a query to the controller. Increasing the number of flows in the data path will eventually bring the controller down to fulfill the switch controller flow install request [1].

Possible Solutions:

DoS mitigation takes three phases of defense: Detection, response, and recovery.

When FloodGuard detects a saturation attack, it stops data from control plane saturation and all table-miss packets in the OpenFlow switch will be redirected to the data plane cache.

SDN-Guard is also a proposed solution. SDN-Guard is considered an SDN application that can be plugged on top of any SDN controller.

H. Man in The Middle Attacks.

SDN relies on the separation between planes, which means there are a lot of links between network nodes, such as: Links between the application plane and control plane, Links between multiple controllers, and other network Links. MINM attack is to insert an agent gadget among network nodes to interrupt data communication and operate on it without knowing any communicating party [15]. MINM attacks have different types, including DNS spoofing, session hijacking, port mirroring, and other forms of attack.

Possible Solutions:

Transport-Layer Security (TLS) was applied to ensure switch-controller secure communication.

FlowChecker and FortNox are also alternative countermeasures that have been proposed to a MITM.

I. Black Hole.

Black hole is a network condition where the flow path ends unexpectedly and the traffic cannot reach the destination. Malicious switches in the flow path can drop or spoof packets, preventing the flow from reaching its destination [12].

Possible Solutions:

SPHINX [6] is a switch black hole detect mechanism. By verifying the flow graph and capturing the flow patterns of the actual network traffic, it can detect if a black hole attack is associated. It also monitors the byte statistics per flow at each switch in the flow path, then it determines if the values of bytes transmitted are more inconsistent than expected.

J. Eavesdropping Attacks.

An Eavesdropping attack is a form of a sniffing or a snooping attack. In SDN, eavesdropping attacks can occur on the data plane, or through the communication channel between the controller on the control plane and the forwarding device on the data plane. In the data plane, eavesdropping attacks can occur on two different levels: the switches themselves or the forwarding links between the switches. Once they are compromised and intercepted, the malicious eavesdroppers can listen to data used for further attacks [4].

Possible Solutions:

The multipath method solves the eavesdropping attack on the communication link between the forwarding devices in the SDN data plane [8].

Hybrid countermeasure to ensure the integrity of the Link Layer Discovery Protocol Packets (LLDP) and to detect and stop any fake link launched by a powerful attacker.

IV. PROPOSED CLASSIFICATION FOR THREATS AND POSSIBLE SOLUTIONS.

In this section, we introduce a proposed classification of SDN threats and attacks. In the previous section, we studied all major SDN attacks. Scientists and researchers tried to study these attacks and they proposed possible solutions for each attack. What we are trying to do in this paper is to collect all the researchers' work in the SDN security field and propose a new classification for all attacks. After we collected data from research papers about SDN attacks counter measurements, we will be classifying them in this chapter. This is to facilitate finding all the research data in one paper for anyone who wants to learn about the SDN security field. Each type of attack targets the network from certain points in one or different planes. In our paper, we introduced all SDN network attack points, any type of attack must reach the network through one of these points. Figure 2 Shows SDN network attacking points.

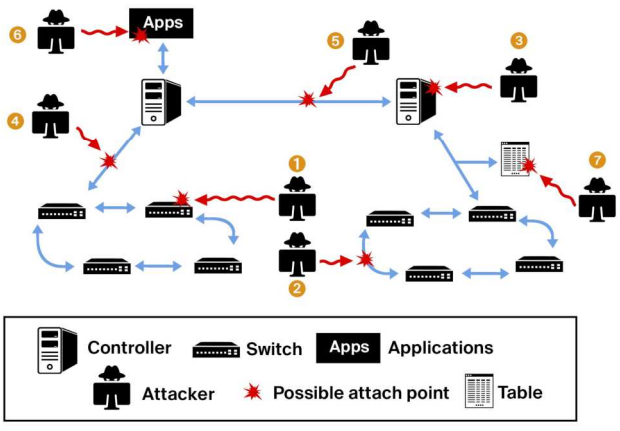

Fig. 2. SDN Network Attacking Points.

SDN Threat	Attacking Points
Lack of Authentication and Authorization	-Applications
Lack of Access Control and Accountability	-Applications -Link applications and controller
Malevolent Applications	-Applications -Controller
Rules Insertion Manipulation	-Link Controller-Switches -Link Applications and Controller
Scalability and Availability	-Link Controller and Switches -Link Applications and Controller
Flooding Attacks	-Controller -Link Apps and controller -Link Controller and Switches -Switches -Flow Table
DoS and DDoS Attacks	-Controller -Link Controller and switches -Switches
Man in the Middle Attacks	-Controller -Link Apps-controller -Link Controller and Switches -Links among switches -Links controllers and Switches
BLACK-HOLE	-Link Controller and Switches -Link Applications and Controller -Links Controller and switches
Eavesdropping Attacks	-Link Controller and Switches -Switches -Links Controller and switches

TABLE I
CLASSIFICATION FOR SDN THREATS AND ATTACKING POINTS

V. DEVELOPED OMNET++ SDN PLATFORM.

In this Section, we tried to develop an SDN model simulation. So, after studying in the previous chapters about SDN history and architecture, and after reviewing scientists' work in the practical and simulation field, as well as studying scientists' work on SDN networks and controller performances, in addition to reviewing challenges and security threats facing SDN, we attempted to develop an SDN simulation platform to test our SDN network and the security threats we studied.

The field of SDN is still rising and has a lot of aspects to study and test. In this chapter, we aim to develop a model that can be used to simulate SDNs. We aim to achieve that by using OMNeT++, which is a network simulation tool, alongside extending the INET framework in OMNeT++ to study multiple network topologies and compare it to SDN. In this section, we simulate, explain and review the SDN OpenFlow network model. We explain the model and how networks run, and how OpenFlow switches and controllers integrate, in addition to explaining the network nodes inside network devices such as controllers and switches, we also review network setup and requests between the OpenFlow controller and switches.

A. *OpenFlow Extension and INET.*

INET Framework is an open-source model library for the OMNeT++ simulation environment. It provides protocols, agents, and other models for researchers and students working with communication networks. INET is especially useful when designing and validating new protocols, or exploring new or exotic scenarios. INET contains models for the Internet stack (TCP, UDP, IPv4, IPv6, OSPF, BGP, etc.), wired and wireless link layer protocols (Ethernet, PPP, IEEE 802.11, etc), support for mobility, MANET protocols, DiffServ, MPLS with LDP and RSVP-TE signaling, several application models, and many other protocols and components. Several other simulation frameworks take INET as a base, and extend it into specific directions, such as vehicular networks, overlay/peer-to-peer networks, LTE or what we will use and that is the OpenFlow extension to simulate SDNs [16].

In INET, other frameworks take it as a base for their extensions. The OpenFlow SDN is a simulation model of the OpenFlow system for INET-3.6 and OMNeT++ 5.6, based on the OpenFlow switch specification 1.2. presented at the 6th International Workshop on OMNeT++ (Cannes, 2013). OpenFlow extension is currently the most famous method in implementing the SDN concept and provides a high degree of flexibility in routing network flows [9]. The development of the OpenFlow standard is managed by the Open Networking Foundation (ONF). The ONF promotes the development and use of Software Defined Networking technologies and OpenFlow is an example.

B. *Model and Environment.*

In order to simulate the SDN environment, we are using the OMNeT++ application. There are different versions of the OMNeT++ application on the OMNeT++ website: https://omnetpp.org. There are also multiple versions for different operating systems. The advantage of the OMNeT++ INET framework 3.6 is that it has many built-in models that can be customized, as it also makes it easier to integrate new models. INET Framework is an open-source model library for the OMNeT++ simulation environment. The INET Framework has implemented an extension for OpenFlow in order to make it modeled. This network represents the OpenFlow network. This network connects a Client (Sender) to an OpenFlow switch, the OpenFlow switch is connected to another OpenFlow switch and connected to the network controller, the network controller is also connected to the second OpenFlow switch, then the second OpenFlow switch is connected to a receiver (server) figure 3. The client and the server are also standard hosts pre-implemented and extended from the INET framework.

VI. CONCLUSIONS

Due to the separation of control and data planes and the programmability of SDN networks, SDN appears to be the most attractive evolutionary structure of future networks. Despite the impressive advantages, SDN still faces many challenges, especially when it comes to network security issues. In this case,

we tried to detect security challenges and threats by analyzing core network architecture, which led us to survey security-conscious network measures in SDN.

In this paper, we firstly reviewed the SDN structure and characteristics of the OpenFlow protocol. Then, we analyzed the current security-based network situation and believed that SDN network management technology was a reliable and effective way to solve these problems. So, we have introduced

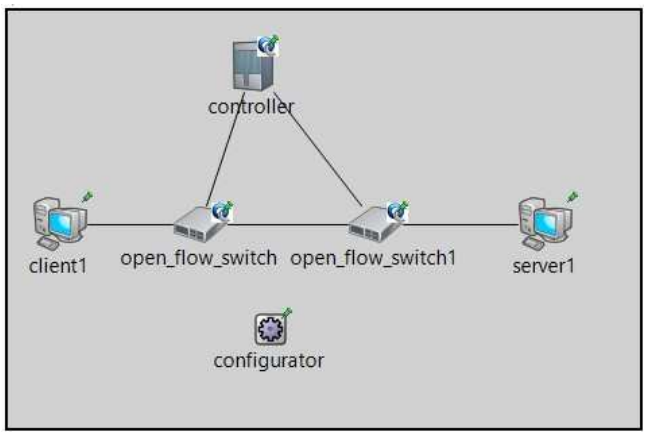

Fig. 3. OpenFlow Network Model.

the security weaknesses and advantages of SDN. Then we highlighted security vulnerabilities at each of the three planes and each one of the seven attack points. Afterwards, we presented possible security solutions for each type of attack. We also summarized the security technologies that can strengthen the security of the SDN network.

Our survey concluded that the most vulnerable part of the SDN architecture is the network controller. So, we researched it from different directions, including scalability and availability and DoS and DDoS attacks. We also concluded that most of the proposed solutions by researchers are software-developed security platforms. And although secure applications have been developed and deployed, the security of the application plane itself is a security challenge.

The field of SDN is still a rising field in the world of networking. This means that new security threats are very likely to emerge with the gradual deployment of SDN technology. Likewise, the threat space may grow, as existing security threats in traditional networks will spread along with SDN-specific security challenges. However, SDN relies on innovation and programmability, which means that an automatic safety mechanism will be developed to achieve rapid anomaly detection and rapid response protection.

VII. FUTURE WORK.

In the future, based on cloud computing, network virtualization and middlebox are to be considered important applications of SDN, and the security threat is yet to further increase. Therefore, the security issues of these applications are expected to attract more and more attention. In our paper, we also introduced a developed OMNeT++ SDN model. This model aims to help researchers study and evaluate SDN networks in general and security threats and mitigation techniques in specific.

So, after reviewing multiple types of attacks and their possible solutions, future work is to practically test these attacks to evaluate their performance metrics, and to propose new mitigation techniques and ideas for the arising attacks while SDN networks expand. New challenges are arising in the scope of SDN networks, such as Malware-based attacks on SDN and Programmable data planes. We believe that future research in this area needs to better understand how the host interacts with and affects the control plane.

REFERENCES

[1] Ijaz Ahmad, Suneth Namal, Mika Ylianttila, and Andrei Gurtov. Security in software defined networks: A survey. *IEEE Communications Surveys Tutorials*, 17(4):2317–2346, 2015.

[2] James Akerele. Performance analysis of openflow-enabled network topologies: Openflow sdn vs traditional network. 01 2018.

[3] Moreno Ambrosin, Mauro Conti, Fabio De Gaspari, and Radha Poovendran. Lineswitch: Tackling control plane saturation attacks in software-defined networking. *IEEE/ACM Transactions on Networking*, 25(2):1206–1219, 2017.

[4] Ahmad Aseeri, Nuttapong Netjinda, and Rattikorn Hewett. Alleviating eavesdropping attacks in software-defined networking data plane. In *Proceedings of the 12th Annual Conference on Cyber and Information Security Research*, CISRC '17, New York, NY, USA, 2017. Association for Computing Machinery.

[5] Normaziah A. Aziz, Teddy Mantoro, M. Aiman Khairudin, and A. Faiz b A. Murshid. Software defined networking (sdn) and its security issues. In *2018 International Conference on Computing, Engineering, and Design (ICCED)*, pages 40–45, 2018.

[6] Mohan Dhawan, Rishabh Poddar, Kshiteej Mahajan, and Vijay Mann. Sphinx: Detecting security attacks in software-defined networks. 01 2015.

[7] Marcial P. Fernandez. Comparing openflow controller paradigms scalability: Reactive and proactive. In *2013 IEEE 27th International Conference on Advanced Information Networking and Applications (AINA)*, pages 1009–1016, 2013.

[8] Eduardo Germano da Silva, Luis Augusto Dias Knob, Juliano Araujo Wickboldt, Luciano Paschoal Gaspary, Lisandro Zambenedetti Granville, and Alberto Schaeffer-Filho. Capitalizing on sdn-based scada systems: An anti-eavesdropping case-study. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 165–173, 2015.

[9] Dominik Klein and Michael Jarschel. An openflow extension for the omnet++ inet framework. pages 322–329, 01 2013.

[10] Diego Kreutz, Fernando M. V. Ramos, Paulo Esteves Ver´ıssimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76, 2015.

[11] Diego Kreutz, Fernando M.V. Ramos, and Paulo Verissimo. Towards secure and dependable software-defined networks. In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, HotSDN '13, page 55–60, New York, NY, USA, 2013. Association for Computing Machinery.

[12] Trupti Lotlikar. A survey of potential security threats and countermeasures in sdn : An iot enabling technology. 2017.

[13] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: Enabling innovation in campus networks. 38(2):69–74, March 2008.

[14] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74, March 2008.

[15] Azath Mubarakali and Abdulrahman Saad Alqahtani. A survey: Security threats and countermeasures in software defined networking. In *2019 IEEE 2nd International Conference on Information and Computer Technologies (ICICT)*, pages 180–185, 2019.

[16] OMNeT++. Inet framework. https://omnetpp.org/downloaditems/INET.html, 2012.

[17] Hitesh Padekar, Younghee Park, Hongxin Hu, and Sang-Yoon Chang. Enabling dynamic access control for controller applications in softwaredefined networks. In *Proceedings of the 21st ACM on Symposium on Access Control Models and Technologies*, SACMAT '16, page 51–61, New York, NY, USA, 2016. Association for Computing Machinery.

[18] Kevin Phemius, Mathieu Bouet, and J´ er´ emie Leguay. Disco: Distributed´ multi-domain sdn controllers. In *2014 IEEE Network Operations and Management Symposium (NOMS)*, pages 1–4, 2014.

[19] Kevin Phemius, Mathieu Bouet, and J´ er´ emie Leguay. Disco: Distributed´ sdn controllers in a multi-domain environment. In *2014 IEEE Network Operations and Management Symposium (NOMS)*, pages 1–2, 2014.

[20] Phillip Porras, S. Cheung, Martin Fong, Keith Skinner, and Vinod Yegneswaran. Securing the software defined network control layer. 01 2015.

[21] Ying Qian, Wanqing You, and Kai Qian. Openflow flow table overflow attacks and countermeasures. In *2016 European Conference on Networks and Communications (EuCNC)*, pages 205–209, 2016.

[22] Ahmed Sallam, Ahmed Refaey, and Abdallah Shami. On the security of sdn: A completed secure and scalable framework using the softwaredefined perimeter. *IEEE Access*, 7:146577–146587, 2019.

[23] Shin Seungwon and Guofei Gu. Attacking software-defined networks: A first feasibility study. pages 165–166, 08 2013.

[24] Shin Seungwon, Vinod Yegneswaran, Phillip Porras, and Guofei Gu. Avant-guard: scalable and vigilant switch flow management in softwaredefined networks. 11 2013.

[25] Seungwon Shin, Phillip Porras, Vinod Yegneswaran, Martin Fong, Guofei Gu, and Mabry Tyson. Fresco: Modular composable security services for softwaredefined networks. In *In Proceedings of Network and Distributed Security Symposium*, 2013.

[26] Marco Tiloca, Alexandra Stagkopoulou, and Gianluca Dini. Performance and security evaluation of sdn networks in omnet++/inet. 09 2016.

[27] Saro Velrajan. Sdn architecture. https://tech.ginkos.in/2012/12/sdnarchitecture.html, 2012.

[28] Xitao Wen, Yan Chen, Chengchen Hu, Chao Shi, and Yi Wang. Towards a secure controller platform for openflow applications. In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, HotSDN '13, page 171–172, New York, NY, USA, 2013. Association for Computing Machinery.

[29] Xitao Wen, Bo Yang, Yan Chen, Chengchen Hu, Yi Wang, Bin Liu, and Xiaolin Chen. Sdnshield: Reconciliating configurable application permissions for sdn app markets. In *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 121–132, 2016.

[30] Jianfeng Xu, Liming Wang, Chen Song, and Zhen Xu. An effective table-overflow attack and defense in software-defined networking. In *2019 IEEE 44th LCN Symposium on Emerging Topics in Networking (LCN Symposium)*, pages 10–17, 2019.

[31] Changhoon Yoon, Seungwon Shin, Phillip Porras, Vinod Yegneswaran, Heedo Kang, Martin Fong, Brian O'Connor, and Thomas Vachuska. A security-mode for carrier-grade sdn controllers. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, ACSAC 2017, page 461–473, New York, NY, USA, 2017. Association for Computing Machinery.

[32] Menghao Zhang, Guanyu Li, Lei Xu, Jun Bi, Guofei Gu, and Jiasong Bai. *Control Plane Reflection Attacks in SDNs: New Attacks and Countermeasures: 21st International Symposium, RAID 2018, Heraklion, Crete, Greece, September 10-12, 2018, Proceedings*, pages 161– 183. 09 2018.